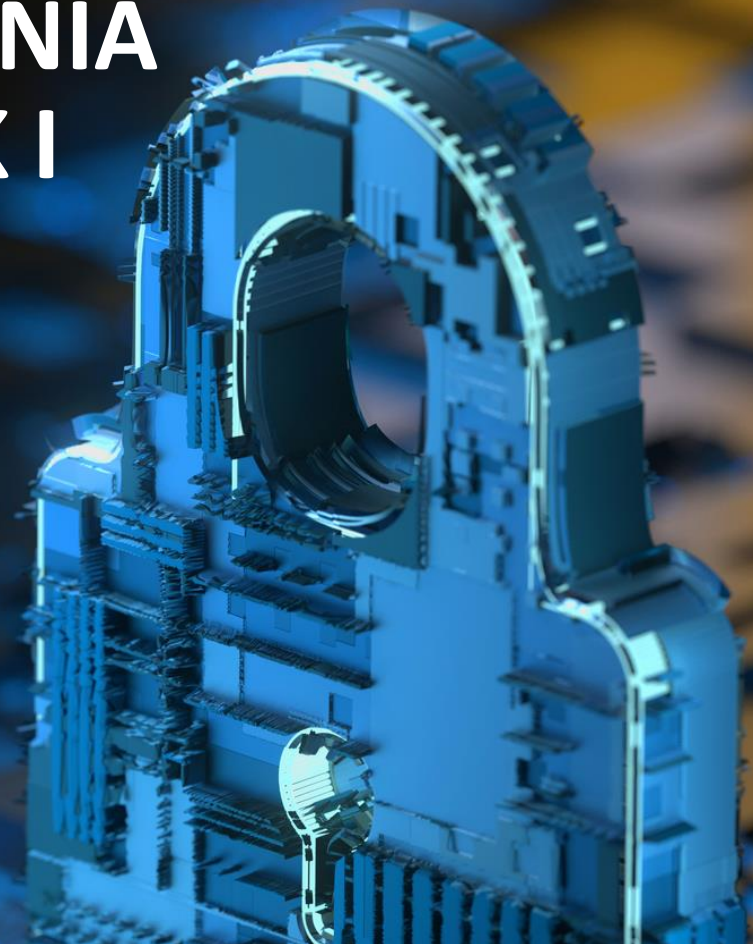
BEZPIECZEŃSTWO KORZYSTANIA Z ZEWNĘTRZNYCH BIBLIOTEK I KOMPONENTÓW

OPEN SOURCE I ZEWNĘTRZNE BIBLIOTEKI W PROCESACH WYTWARZANIA, WDRAŻANIA I UTRZYMYWANIA OPROGRAMOWANIA



FILIP BRANDT

CSO Council, 23.11.2023

BNP PARIBAS

The bank for a changing world

Filip Brandt

Security Architect

Architekt bezpieczeństwa aplikacji w Banku BNP Paribas
Z pracą w cyberbezpieczeństwie BNP Paribas związany od 2020 roku
Odpowiedzialny między innymi za bezpieczeństwo w procesach wytwórczych, wdrożenie Threat Modellingu, inicjatywę Security Champions

AGEND

A

01 Wprowadzenie

02 Kluczowe ryzyka

03 Jak sobie z nimi radzić?

BNP PARIBAS

Wprowadzenie

CZYM JEST OPEN-SOURCE?

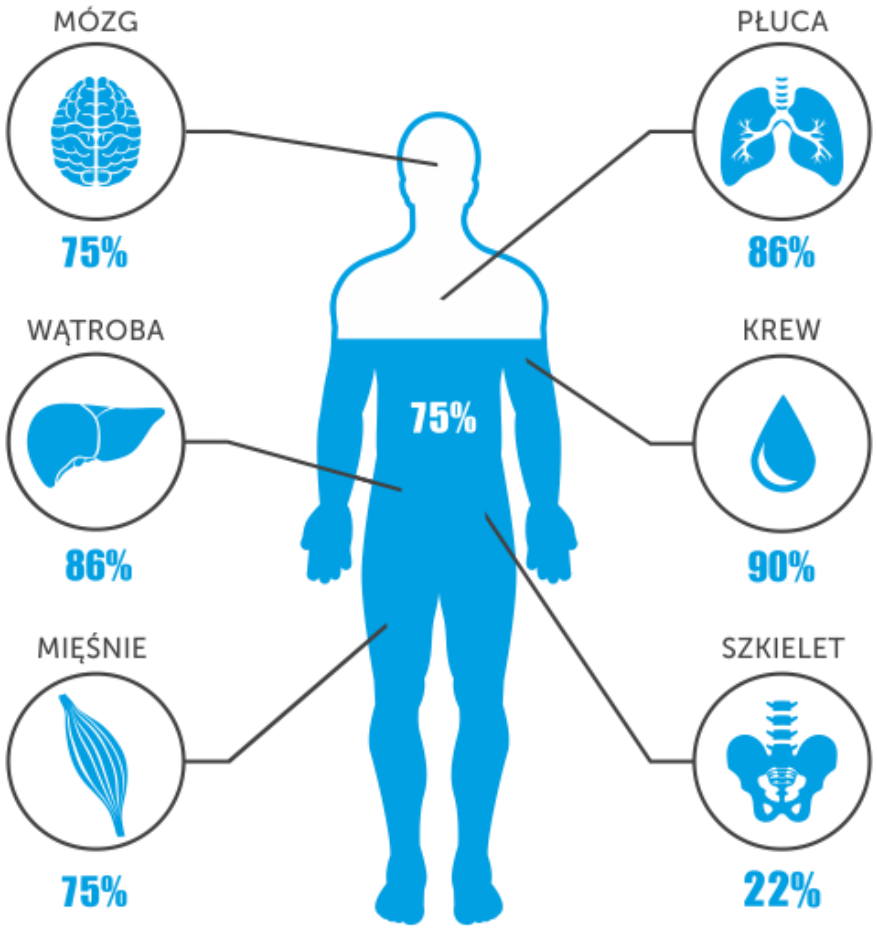
BNP PARIBAS

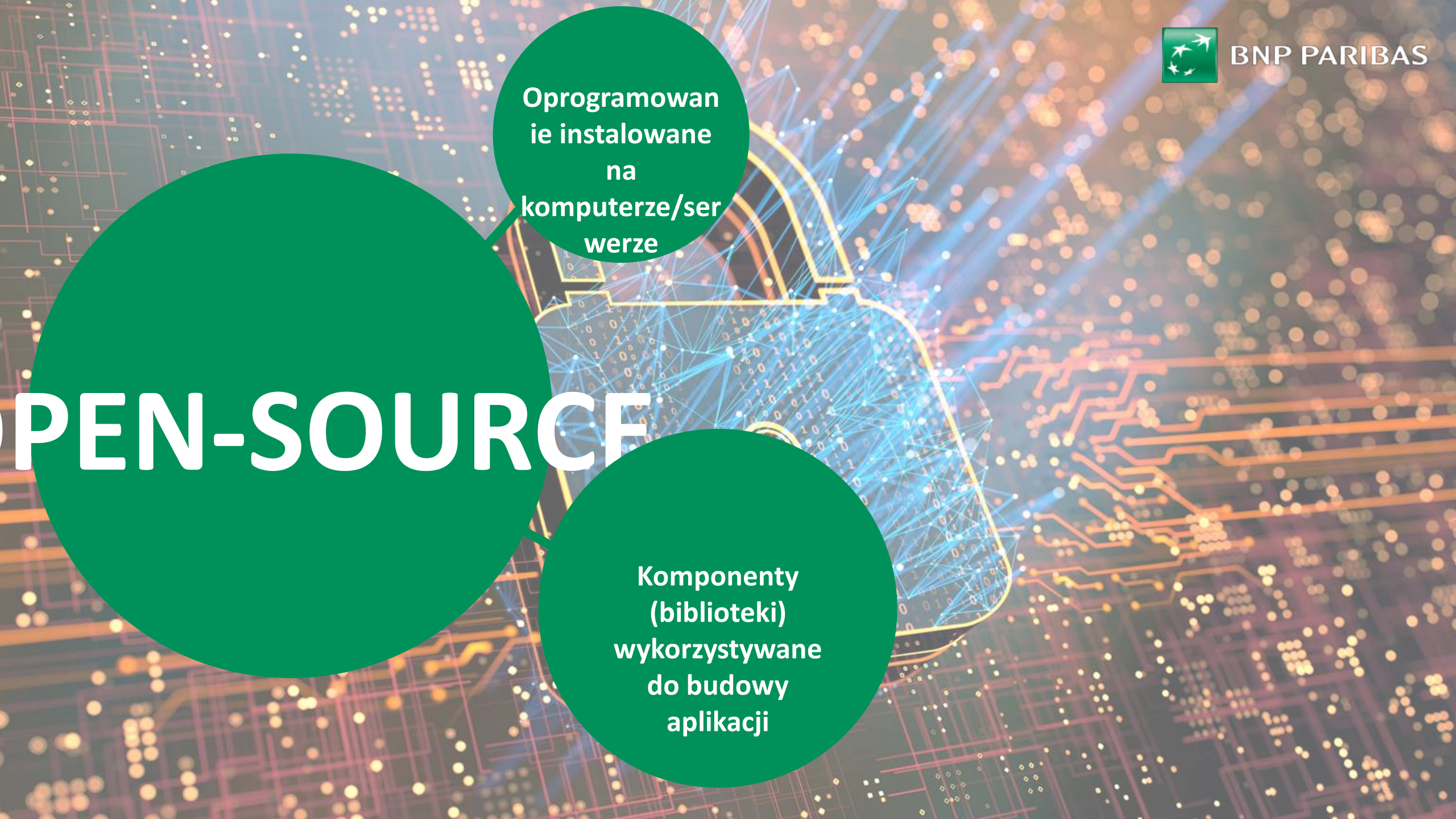
Czym jest oprogramowanie open-source?

Rodzaj oprogramowania, w którym kod źródłowy jest wydawany na podstawie licencji, na mocy której właściciel praw autorskich przyznaje użytkownikom prawa do badania, zmiany i rozpowszechniania oprogramowania w ramach licencji wolnego oprogramowania

Źródło: https://opensource.org/

OPEN-SOURCE – CZY TO DUŻA CZĘŚĆ?




BNP PARIBAS
Oprogramowanie instalowane na komputerze/serwerze
PEN-SOURCE
Komponenty (biblioteki) wykorzystywane do budowy aplikacji

OPROGRAMOWANIE INSTALOWANE NA KOMPUTERZE/SERWERZE

Oprogramowanie pobierane i instalowane na serwerze bądź stacji przez użytkownika, od razu gotowe do działania

Jest to obszar, który jest znany i zarządzany od dłuższego czasu, najczęściej dostrzegany oraz kontrolowany przez organizację

Przykład:
Oprogramowanie do zarządzania urządzeniami peryferyjnymi od dostawców

KOMPONENTY WYKORZYSTYWANE DO BUDOWY APLIKACJI

Oprogramowanie pobierane przez użytkownika, aby następnie zostało odpowiednio zaimplementowane w budowanej aplikacji
Ten obszar często jest marginalizowany bądź wręcz niedostrzegany przez organizacje

Przykład:
biblioteka log4j (log4shell)

Log4Shell

14 languages

Contents [hide]

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

Log4Shell (**CVE-2021-44228**) is a zero-day vulnerability in Log4j, a popular Java logging framework, involving arbitrary code execution.[2][3] The vulnerability had existed unnoticed since 2013 and was privately disclosed to the Apache Software Foundation, of which Log4j is a project, by Chen Zhaojun of Alibaba Cloud's security team on 24 November 2021. Before an official CVE identifier was made available on 10 December 2021, the vulnerability circulated with the name "Log4Shell", given by Free Wortley of the LunaSec team, which was initially used to track the issue online.[2][1][4][5][6] Apache gave Log4Shell a CVSS severity rating of 10, the highest available score.[7] The exploit was simple to execute and is estimated to have had the potential to affect hundreds of millions of devices.[6][8]

Log4Shell

CVE identifier(s)	CVE-2021-44228
Date discovered	24 November 2021; 23 months ago
Date patched	6 December 2021; 23 months ago
Discoverer	Chen Zhaojun of the Alibaba Cloud Security Team[1]
Affected software	Applications logging user input using Log4j 2

The vulnerability takes advantage of Log4j's allowing requests to arbitrary LDAP and JNDI servers,[2][9][10] allowing attackers to execute arbitrary Java code on a server or other computer, or leak sensitive information.[5] A list of its affected software projects has been published by the Apache Security Team.[11] Affected commercial services include Amazon Web Services,[12] Cloudflare, iCloud,[13] *Minecraft: Java Edition*,[14] Steam, Tencent QQ and many others.[9][15][16] According to Wiz and EY, the vulnerability affected 93% of enterprise cloud environments.[17]

The vulnerability's disclosure received strong reactions from cybersecurity experts. Cybersecurity company Tenable said the exploit was "the single biggest, most critical vulnerability ever,"[18] *Ars Technica* called it "arguably the most severe vulnerability ever"[19] and *The Washington Post* said that descriptions by security professionals "border on the apocalyptic."[8]

BNP PARIBAS

Ryzyko

JAKIE WYSTĘPUJĄ ZAGROŻENIA W TYM OBSZARZE?

KLUCZOWE RYZYKA

Złośliwy kod

Rzadkie, ale bardzo niebezpieczne ze względu na potencjalnie wysoki wpływ na bezpieczeństwo

Licencje

Typy licencji i możliwy zakres ich wykorzystania

Podatności

Ze względu na skalę wykorzystania open-source analogicznie rośnie liczba podatności/znalezisk

Inne...

PODATNOŚCI

Skanery infrastrukturalne
Automatyczne skanery bezpieczeństwa wykrywające nieaktualne/podatne biblioteki i usługi na serwerach

RedTeaming
Dodatkowa akcja weryfikacji z perspektywy np. całej infrastruktury organizacji

Skanery BAS (Breach Attack Simulation)
Automatyczne skanery identyfikujące podatne usługi i weryfikujące znane metody exploitacji

Skanery aplikacyjne
Skanery kodu (SAST), skanery weryfikujące podatności w działającej aplikacji (DAST/IAST)

Skanery aplikacyjne
Skanery SCA (Software Composition Analysis), skanery dedykowane do weryfikacji opensource na etapie rozwoju oprogramowania, przed wdrożenie i w trakcie działania

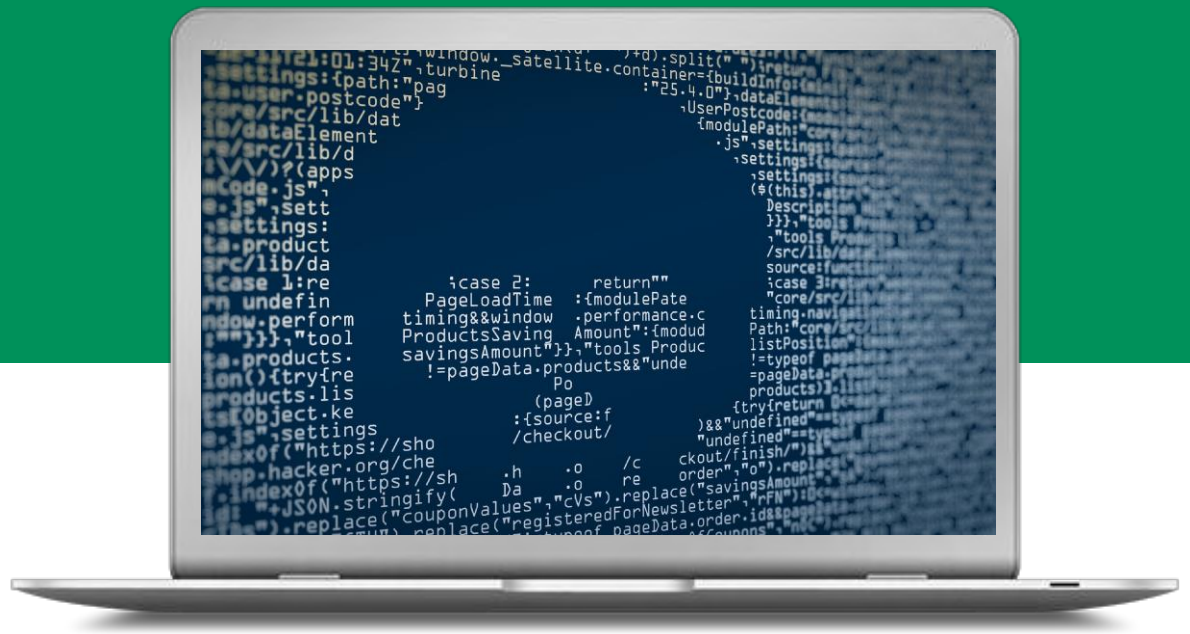
Pentesty
Weryfikacja możliwości exploitacji podatności na żywym organizmie

LICENCJE

Nieznajomość postanowień licencyjnych

W bardzo konkretnych przypadkach wykorzystanie biblioteki może między innymi implikować konieczność udostępnienia kodu źródłowego rozwiązania korzystającego z tej biblioteki

ZŁOŚLIWY KOD W KOMPONETACH OPENSOURCE



Frustracja developera

Znane przypadki ataków na bardzo popularne biblioteki (np. color.js, faker.js)

Ataki na repozytoria

Kolejne przypadki wykorzystania luk w mechanizmach NPM, które pozwolą na pobranie odpowiednio przygotowanej biblioteki (dependency confusion)

5.1

Average number of outstanding, critical vulnerabilities in an application.

Ranges between 2.6 and 9.5 based on programming language.

24%

of organizations are confident in the security of their direct dependencies.

59%

of organizations report their OSS is somewhat or highly secure.

68.8

Average dependencies per project.

Ranges between 25 and 174 based on programming language.

18%

of organizations are confident in the security of their transitive dependencies.

SCA and SAST tools

are the #1 and #2 tools used to address security concerns.

97.8

Average number of days it takes to fix a vulnerability.

49%

of organizations have a security policy that addresses OSS.

73%

of organizations are searching for best practices to improve their software security.

Increased incentives

by employer is the #1 approach to improving OSS resourcing.

More intelligent tools

are the #1 way organizations intend to improve supply chain security.

11%

Average increase to an organization's security score in 2022.

Źródło: Linux Foundation oraz Snyk „Cybersecurity Challenges in OSS”

State of the Software Supply Chain by the numbers

1 in 8 open source downloads have known risk

245,000 malicious packages discovered –2X all previous years combined

18.6% of open source projects across Java and JavaScript that were maintained in 2022, are no longer maintained today

96% of vulnerable downloaded releases had a fixed version available

10 superior versions of components are typically available, for every nonoptimal component upgrade made

2x When paired with optimal upgrades, good data saves you twice as much time or nearly 1.5 months of time per application, per year when upgrading components.

135% increase in the adoption of AI and ML components within corporate environments, over the last year

67% of survey respondents feel confident that their applications do not rely on known vulnerable libraries, despite 10% of respondents reporting their organizations had security breaches due to open source vulnerabilities in the last 12 months

Źródło: Sonatype „9th Annual State of the Software Supply Chain”

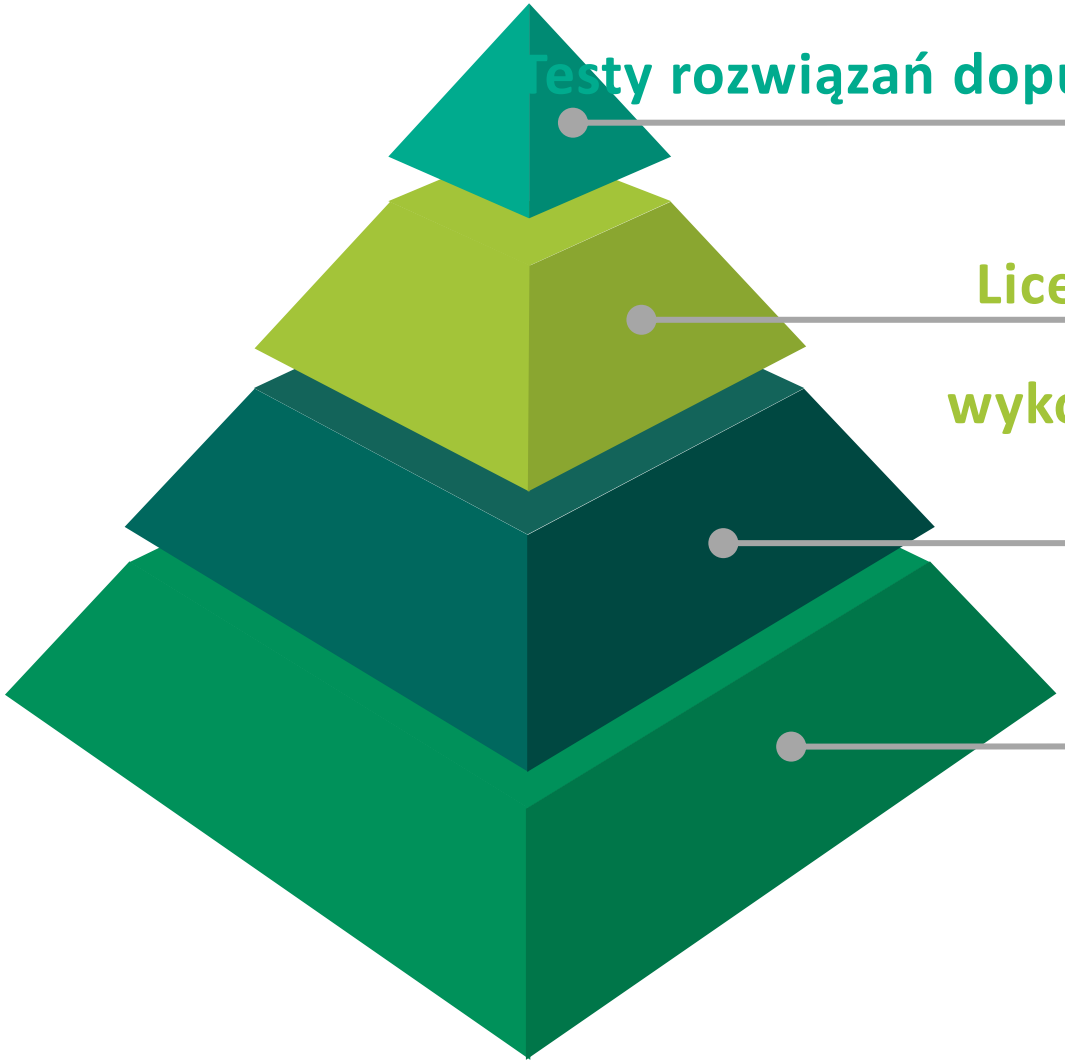
BNP PARIBAS

Postępowanie z ryzykiem

JAK SOBIE Z TYM RADZIĆ?

Oprogramowanie open-source instalowane na serwerach/komputerach

DOPUSZCZENIE OPROGRAMOWANIA DO WYKORZYSTANIA



Testy rozwiązań dopuszczanych do użytku

Licencje i zakres wykorzystywania

Czy jest nadal rozwijana

Bieżące informacje o podatnościach

+ WŁAŚCICIELSTWO

KONTROLA

Repozytoria z oprogramowaniem

Użytkownik może instalować domyślnie tylko to, co ma dostępne w repozytorium

Większa kontrola IT

IT może więcej, więc i odpowiednia kontrola musi być przeprowadzana

Potwierdzenie postanowień licencyjnych

Nowe wersje w teorii powinny posiadać analogiczne zapisy licencyjne, ale warto to potwierdzać, kiedy to możliwe

Weryfikacja podatności i nowe wersje

Wersje oprogramowania muszą być aktualizowane w repozytoriach, ale zawsze po weryfikacji

Komponenty open-source wykorzystywane do budowania oprogramowania

Podejście do komponentów wykorzystywanych do budowy oprogramowania jest znacznie inne, bardziej skomplikowane oraz często nieodpowiednio dostrzegane przez organizacje

Dochodzą tutaj nowe aspekty oraz wykorzystywane narzędzia, które należy uwzględnić zarówno w developmencie (np. proxy, firewalling) jak i świadomości samych odpowiedzialnych (zasady skanowania, zasady wdrożenia na produkcję)

JAK PODCHODZIĆ?

Repozytorium proxy

Firewalling

Dostęp do narzędzi SCA

Analiza przed releasem

APLIKACJE WYTWARZANE POZA ORGANIZACJĄ

Uniwersalne standardy wytwarzania

Niezależnie od tego, czy aplikacja jest wytwarzana wewnątrz organizacji, czy na zewnątrz, powinny zostać przyłożone zunifikowane standardy jej wytwarzania

Aplikacje mikroserwisowe – co z obrazami?

Obrazy budujemy u siebie i korzystamy z bibliotek pobieranych przez nasze repozytoria proxy

Zapewnienie możliwości wdrożenia standardów

W kontaktach z dostawcami szczególnie ważne jest zapewnienie możliwości implementacji standardów zgodnych z przyjętymi w organizacji

Dostarczenie odpowiednich artefaktów

Na etapie uzgadniania warunków dostarczania oprogramowania zapewniamy sobie możliwość wykonania skanu komponentów open-source

TROCHĘ STATYSTYK – SCA W BNP PARIBAS

1600

Obiektów skanowanych w SCA

33000

Liczba ewaluacji na miesiąc

40

Średnio liczba skanów SCA dla aplikacji miesięcznie

200

Średnio komponentów

65

Średnio podatności SCA per ewaluacja

9

Różnych ekosystemów skanowanych

160

Aplikacji skanowanych w SCA

ROZWIĄZANIA PUDEŁKOWE? SBOM?

A co w przypadku wdrażania rozwiązań pudełkowych?

Zawsze warto dopytać czy dostawca produkuje SBOM dla swojego soft i próbować weryfikować każdą wersję, wyjaśniając wątpliwości w zakresie opensource

A co jeśli tego SBOMa nie ma?

Powinniśmy albo wręcz musimy szukać rozwiązań dla podatności wprowadzanych przez opensource wykorzystywanych w takich rozwiązaniach

The SBOM is a Component of SSCS

"In the end, the trust we place in our digital infrastructure should be proportional to how trustworthy and transparent that infrastructure is ..."

Executive Order 14028 — 12 May 2021

"SBOMs are a critical step toward securing the software supply chain."

The Minimum Elements for a Software Bill of Materials — U.S. Department of Commerce

Gartner

Security of the software supply chain is now as critical as the security of the software itself.

Gartner

Of organizations surveyed,
98% use open source software.

FIGURE 6 | PAGE 15

Of organizations surveyed,
95% are concerned about software security.

FIGURE 9 | PAGE 18

Of organizations surveyed,
76% currently have a level of SBOM readiness.

FIGURE 20 | PAGE 29

Of organizations surveyed,
47% are using SBOMs today.

FIGURES 21 & 24 | PAGE 30

SBOM use will increase by 66%
for organizations in 2022.

FIGURE 29 | PAGE 43

Based on organizations surveyed, it's forecasted
78% will use SBOMs in 2022.

FIGURE 29 | PAGE 43

Based on organizations surveyed, it's forecasted
88% will use SBOMs in 2023.

FIGURE 29 | PAGE 43

Of organizations using SBOMs today,
74% produce AND consume SBOMs.

FIGURES 21 & 24 | PAGE 30

#1 ACTION:
Get a vulnerability reporting system
in order to better secure your software supply chain.

FIGURE 13 | PAGE 23

#2 ACTION:
Use SBOMS to better secure your software supply chain.

FIGURE 13 | PAGE 23

When *producing* SBOMs...
#1 BENEFIT:
developers better understand dependencies.

FIGURE 22 | PAGE 31

When *consuming* SBOMs...
#1 BENEFIT:
better support for compliance and reporting.

FIGURE 25 | PAGE 37

CYBER RESILIENCE ACT VS SBOM

Temat SBOMa nie jest również obcy w Europie i znajduje swoje ważne miejsce również w dokumencie Cyber Resilience Act

Przewidziano w nim kilka ważnych paragrafów dedykowanych tematowi open-source i warto śledzić postanowienia zawarte na przykład w poniższych paragrafach ze względu na ich potencjalnie bardzo duży wpływem na wykorzystywanie open-source w Europie:

- Paragraf 4
- Paragraf 6
- Paragraf 10
- Paragraf 12
- **Artykuł 37:**

„A software bill of materials can provide those who manufacture, purchase, and operate software with information that enhances their understanding of the supply chain, which has multiple benefits, most notably it helps manufacturers and users to track known newly emerged vulnerabilities and risks. It is of particular importance for manufacturers to ensure that their products do not contain vulnerable components developed by third parties.

SBOM in the CRA

- **Manufacturers to draw up an SBOM** in a commonly used format covering at the very least the top-level dependencies of the product
- **No requirement** to make the SBOM publicly available
- SBOM to be included in the **technical documentation** and, upon request, to be provided to **market surveillance authorities**
- **Commission empowerment** to specify the format and elements (international standards to be relied upon)

THANK merci

mèsita **YOGRA** chokrane

dhanyavad **ZARIG** dziękuję

GRACIAS danke **ATÔ** **NANDRI**

дякую **MAH** teşekkür

ederim **AIO** **JËRËJËF**